REMARKS

The Examiner is thanked for the thorough examination of this application. The Office Action, however, has tentatively rejected all claims 1-25. For at least the reasons set forth herein, Applicant disagrees and requests reconsideration and withdrawal of the rejections.

The specification is amended, above, to include the patent number of the recently-issued patent (also invented by the inventor of the present application) which was referred to by serial number in the present application.

The Office Action objected to claims 5, 6, 13, 14, and 15. The undersigned does not understand the basis for this objection. As to claims 5 and 6, both of these claims depend from claim 2. Intervening claim 3 depends from claim 2, and claim 4 depends from claim 3 and thus depends through claim 3 on claim 2 as well. Therefore the undersigned submits that there is nothing improper with the numbering and order of claims 5 and 6. With regard to claims 13, 14, and 15, the undersigned has amended claims 13 and 14 so that those claims depend from claim 12 (which they were initially intended to depend from). As amended, the undersigned submits that the ordering and numbering of these claims is proper. Notwithstanding, if the Examiner maintains the rejections, the undersigned will be happy to cancel all claims and submit an entirely new set of substantively-identical (but renumbered) claims.

Discussion of Substantive Rejections

Fundamental Distinction of Cited References

Before addressing the specific rejections, Applicant first notes that there are certain fundamental distinctions between the cited references and the claimed embodiments of the invention. For example, the principal reference relied upon by the Office Action (U.S. Patent

6,539,381) is directed to the synchronization of database information. Specifically, the '381 patent is directed to the synchronization of replicas of distributed databases. In contrast, claimed embodiments of the present invention are directed to the maintenance, when appropriate, of server-side state of interest to a single user in systems having multiple, collaborating servers. Thus, claimed embodiments are directed to the proper maintenance of single-user state information, such as user session state, and even then only when appropriate. The Office Action doesn't seem to appreciate this significant distinction. The claim language, which embodies this distinction, will be discussed in more detail below.

To illustrate the relevance of this distinction, consider a user of a Web application distributed across multiple Web servers (or clusters of servers), where the servers/clusters do not inherently replicate server-side state (e.g., session state) with one another, but where the servers/clusters are interested in the same kinds of state information about a user. For example, consider an on-line shopping store that is implemented through multiple servers/clusters. Assume a user (shopper) logs-in at one server/cluster and then links across to another server/cluster to do some shopping. The latter server/cluster is interested in the user's login state - whether the user is logged-in and, if so, the user's registered identity (e.g., name, email address, etc.). Knowing this information allows the second server/cluster to prevent access without login, and, with login, allows it to display an appropriate greeting (e.g., "Welcome <name>") to the user who is logged-in. However, the server-side state of the former server/cluster is not inherently (e.g., by the Web application server software) replicated to the latter. This may be due to performance reasons, or perhaps for compatibility reasons (e.g., the 2 servers/clusters may be implemented using different Web application server software, which are natively incapable of mutual server-side state replication). This creates a problem.

Embodiments described in now-issued patent 6,845,390 provide one solution by taking advantage of the fact that the two servers/clusters do not, in fact, need to replicate one another at all, unless (and until) the user actually navigates between them. After all, if the user stays on the first server/cluster the whole time, the state on the second server/cluster need not be updated. Therefore, replication is only needed if (and no sooner than when) the user actually navigates between the servers/clusters. Embodiments of the 6,845,390 patent have the first server/cluster set some information (e.g., a cookie) on the user's client computer, which indicates that state has changed. If the user never points his client computer to the second server/cluster, no state is replicated (and there is no harm done by the failure to replicate). However, if the user does navigate to the second server/cluster, then information on the user's client is passed by the client up to the second server/cluster. This passed information informs the second server/cluster that state information on the first server/cluster has changed. Therefore, at that point, the second server/cluster does something to update its state (e.g., calls the first server/cluster and retrieves fresh state information which it stores locally). Finally, the second server/cluster saves the information passed to it by the client, so that it will remember it has seen that "signal" already, and thus not needlessly re-update itself again.

Embodiments of the present application define additional solutions to the server-side state issue (see e.g., embodiments described in the specification on pages 29-72). These embodiments expand on an embodiment of the '390 patent, by utilizing information (e.g., a cookie) passed via the client computer between the servers/clusters, as (and if) the user so navigates. However, the information passed is different, and thus the way of digesting the information, recognizing the local state is stale, and updating it, is different. Specifically, embodiments of the present invention don't have the first server/cluster record information

indicating that certain state has changed. Instead, they have it record information indicating that the user has performed certain events locally. For example, instead of indicating that the login state on the first server/cluster has changed (like embodiments of the 6,845,390 patent), embodiments of the present invention indicate that the user performed a login operation (e.g., event). Similarly, when the user logs-out, instead of indicating that the login state has been reset, embodiments indicate merely that the user performed a logout event.

After the second server/cluster notices these new events, it repeats them locally (and automatically). That is, the second server/cluster may do whatever the event entails (it depends on the event and the architecture of the system in regard to the event and the state affected by the event) and does not necessarily need to contact the first server/cluster (or any other server/cluster) to get fresh data or server-side state information, unlike embodiments of the '390 patent. This alternative embodiment enjoys an advantage in that it further abstracts the state change so that servers (which could be of quite different architectures) can have their actual state representations completely decoupled from one another (see discussion in specification pp. 29-35).

Simply stated, the cited references in the Office Action do not disclose or suggest features relevant to embodiments of the present invention. For example, the Prasad reference ('381 patent) relates to database replication between servers that cannot directly communicate with one another. Each server knows about the other servers in the pool and their states, so each server knows - when it receives an update - which other servers need to be contacted with the update, whether directly or indirectly (i.e. by relay). The server communicates with those servers that can be directly contacted, and delegates to them the task of further updating the servers which are unreachable by the first. When servers update themselves with new

replicas, they communicate their updated state back to the sending server, so that a server that communicates-out an update gets back an acknowledgement of the update from each server.

There are a number of differences between embodiments of the present invention and the teachings of Prasad. First, the Prasad invention is a technique for replicating data irrespective of individual user behavior and hence irrespective of individual user need. When the Prasad system records an update, it functions to replicate that update out to the other replica servers in the pool without considering whether the replication is needed. This may make sense in a distributed database, which typically has so many users that no one user, or indeed any definite set of users, can be said to have exclusive interest in the data. The very nature of databases is that they record information which typically is of interest to many users - for example, a library system in which an individual book record is of interest to the librarian, purchaser, check-out counter, and limitless numbers of library customers. Thus, in such an environment it can be assumed that "somebody" will need the data to be at each replica of the database, and thus a replication system like Prasad's - which replicates data deterministically whether it ever actually gets used at each replica site or not - is suitable.

However, in the context of Web applications and user state stored server-side in such applications, which deal with state that is only of interest to one particular user - a user who necessarily employs a client computer to browse the application - the client computer is capable of functioning as an interchange or relay for propagating state-change signals between servers/clusters in the application. Thus, embodiments of the present invention can take advantage of this to do something fundamentally different than Prasad: implement unpredictable "only-as-needed" or "just-in-time" replication of state. Indeed, the deterministic "broadcast" approach of Prasad (in which every server in the pool is updated) is similar to the admitted prior art disclosed on page 9 of the present application.

There are other distinctions between embodiments of the present invention and the cited art, which need not be described herein, as they are superfluous to the differences that are described herein.

Discussion of Specific Rejections

The Office Action rejected independent claim 1 under 35 U.S.C. § 102(e) as allegedly anticipated by U.S. Patent 6,539,381. For at least the reasons set forth below, Applicant submits that the rejection is misplaced and should be withdrawn.

Independent claim 1 (as currently amended) recites:

> 1. A method of replicating server-side state information among a plurality of collaborating servers connected to a network, the method comprising:
> ***determining at a subscriber server from information stored on a client computer whether an event has been performed on a publisher server at the request of the client computer, which event implicates a need for state change on the subscriber server;*** and
> if such an event has been performed, ***replicating state effects of the event*** into state on the subscriber server.

(*Emphasis added*). At least the language emphasized above, in view of the discussion provided above, clearly defines over the cited '381 patent. In this regard, the Office Action (page 3) cited col. 3, lines 41-56 of the '381 patent as allegedly anticipating this claim. Applicant respectfully disagrees. The cited portion of the '381 patent actually states:

> In another aspect, a method for synchronizing replicas of a distributed database is provided. The replicas forming a replica set, wherein each replica of the replica set is stored on one of a plurality of servers in a network. The method comprises steps of storing, at a first server, a plurality of timestamps associated with a plurality of replicas located on each of the plurality of servers responsive to a change in a local replica at the first server, the change is transmitted to a second server, wherein transmission of the change depends upon a comparison of a timestamp of the replica of the first server and a timestamp of the replica of the second server. The method further comprises a step of updating, at the second server, the replica of the second server to reflect the change. The method further comprises a step of storing, at the second

> server, a new timestamp indicating a time at which the replica of the second server was last updated.

As can be readily verified from even a cursory reading of this cited portion of the '381 patent, the '381 patent fails to disclose or suggest the invention defined by claim 1.

As more amply described above, this portion of the '381 patent confirms its operation of synchronizing replicas of a distributed database. It achieves this operation by storing "a plurality of timestamps ... in a local replica at the first server", and transmitting database changes to a second server based "upon a comparison of a timestamp of the replica of the first server and a timestamp of the replica of the second server." Thus database change is propagated through the plurality of servers based on which server is most up-to-date, and irrespective of any particular user need as directed by a client computer. In contrast (and in addition to other distinctions), claim 1 defines an action of "***determining at a subscriber server from information stored on a client computer whether an event has been performed on a publisher server at the request of the client computer, which event implicates a need for state change on the subscriber server.*** (*Emphasis added.*) Portions of this defining step are emphasized to assist the Examiner in appreciating the significant distinctions of this claimed invention over the teachings of the '381 patent. These distinctions include storing information, at the client (*i.e.*, user) computer, which is used by a subscriber server (e.g., first server/cluster) to determine whether the user has previously performed an "event" (e.g., login) on a publisher server (e.g., second server/cluster) that implicates a need to update the state of the user on the subscriber server. Obviously, if no such event has occurred, or if it did occur at the second server/cluster but the user never subsequently pointed his or her client computer to the first, then no state update is needed to the first, and none occurs. Thus state change is

replicated in "as-needed" fashion, based on user navigation via the client computer. This alone clearly defines claim 1 over the '381 patent.

For at least this reason, the rejection of claim 1 is misplaced and should be withdrawn. Claims 2-21 each depend, in some fashion, from claim 1 and therefore define over the cited art for at least the same reasons as claim 1. In addition, these claims further define features that are not taught or suggested in the '381 patent, and these additionally-distinguishing features should be appreciated from the foregoing discussion of fundamental distinctions between embodiments of the present invention and the '381 patent.

The Office Action tentatively rejected independent claim 22 as allegedly obvious over the combination of the '381 patent and U.S. patent 6,549,916 to Sedlar. For at least the reasons set forth herein, Applicant disagrees. Independent claim 22 (as currently amended) recites:

> ***22. A system for replicating server-side state information among a plurality of collaborating servers connected to a network, the system comprising:***
> ***logic configured to determine at a subscriber server from information stored on a client computer whether an event has been performed on a publisher server at the request of the client computer, which event implicates a need for state change on the subscriber server;***
> and
> logic configured to replicate state effects of the event into state on the subscriber server, if such an event has been performed.

(*Emphasis added.*) The element emphasized above loosely corresponds to the element that was emphasized in claim 1, and distinguished from the '381 patent above. Therefore, Applicant submits that claim 22 defines over the cited art for reasons similar to those advanced in connection with claim 1.

In addition to col. 3, lines 41-54 (quoted above in connection with claim 1), the Office Action also cites col. 19, line 64 through col. 20 line 33 and col. 22, lines 9-55 of the '381

patent as allegedly teaching the element emphasized above. These additionally-cited portions of the '381 patent comprise portions of the claims, but do not identify any teachings that are relevant to the claimed element.

Further, it appears that Sedlar has been cited only for the alleged teaching of event notification. Moreover, the Office Action has alleged that a person of ordinary skill would have been motivated to combine the teachings of Sedlar with the '381 patent because the "system would be desirable in that the file system event notification mechanism allows a file cache to be proactively updated so that it always reflects the current state of the files at their original locations." As amply described above, embodiments of the present invention are not directed to replicating and synchronizing data or information across a distributed database. Instead, they are merely to maintain server-side state for a single user when and where needed. As such, even the combined teachings of the cited art do not teach the invention of claim 22.

For at least the foregoing reasons, Applicant respectfully requests that the rejection of independent claim 22 be withdrawn. For at least the same reasons, the rejections of dependent claims 23-25 should be withdrawn as well.

CONCLUSION

Applicant respectfully submits that all claims are now in proper condition for allowance, and respectfully request that the Examiner pass this case to issuance. If, in the opinion of the Examiner, a telephonic conference would expedite the examination of this matter, the Examiner is invited to call the undersigned attorney at (770) 933-9500.

No fee is believed to be due in connection with this Amendment and Response to Office Action. If, however, any fee is deemed to be payable, you are hereby authorized to charge any such fee to Hewlett-Packard Company's Deposit Account No. 08-2025.

Respectfully submitted,

By: 

Daniel R. McClure
Registration No. 38,962

770-933-9500

Please continue to send all future correspondence to:

Hewlett-Packard Development Company, L.P.
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400